METHOD AND SYSTEM FOR LIMITING THE CARDINALITY OF AN SQL QUERY RESULT

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates generally to database management system query engines, and more particularly to enhancing the efficiency of query processing in a database management system by limiting the query result by a query-specified cardinality.

2. Description of the Related Art

A relational database management system (RDBMS) is a computer database management system that uses relational techniques for storing and retrieving data. Relational databases are computerized information storage and retrieval systems in which data in the form of tables (formally, "relations") are typically stored for use on disk drives or similar mass data stores. A "relation" includes a set of rows (formally, "tuples" or "records") spanning several columns (formally, "attributes"). A "tuple" expresses a mathematical relation between its "attributes" (column elements), while a "record" does not. Reference is made to C. J. Date, *An Introduction to Database Systems,* 6th Ed., Addision-Wesley, Reading, M. A. (1994) for general treatment of the relational database art.

An RDBMS is structured to accept commands to store, retrieve and delete data using high-level query languages such as the structured query language SQL (SQL). The term "query" denominates a set of commands for retrieving data from a stored database. The SQL standard has been promulgated by the International Standards Association since 1986. Reference is made, for example, to the SQL-92 Standard "Database Language SQL" published by the ANSI as ANSI X3.135–1992 and published by the ISO as ISO/IEC 9075:1992 for the official specification of the 1992 version of the Structured Query Language.

As used herein, a "query" refers to a set of user commands for retrieving data from a stored database. SQL is used to communicate queries to an RDBMS. SQL requires the return of a particular result set in response to a particular query, but the method of query execution ("Query Execution Plan") employed by the RDBMS is not specified by the query. There are typically many different useful execution plans for any particular query, each of which returns the required result set. For large databases, the execution plan executed by the RDBMS to execute a query must provide the required data return at a reasonable cost and time and hardware resource. Almost all RDBMSs include a query optimizer to translate queries into an efficiently executable plan. Query compilation and optimization for SQL are disclosed in detail in U.S. Pat. Nos. 5,367,675, 5,546,576 and 5,546,570, all assigned to the assignee of this application and incorporated in their entirety by this reference, and by U.S. patent application Ser. No. 08/394,532, filed Feb. 17, 1995 (now U.S. Pat. No. 5,619,692, issued, Apr. 8, 1997), which is also assigned to the assignee of this application and incorporated in its entirety by this reference.

Queries are submitted by users of the RDBMS. As understood in the art, the term "user" has a manifold meaning that encompasses, for example, a human operator, an application program, a remote machine, and so on.

As one example, a database might contain business data, and a user might want to know the past week's gross rental income for rental videos, ordered by income.

As another example of a query of a database, a user might want the RDBMS to retrieve a result set comprising images that resemble an input image, possibly ordered by resemblance, where resemblance is determined by a user-defined function on the image data type.

As recognized by the present invention, while highly effective as a database search language, it happens that SQL makes no provision for explicitly limiting the size ("cardinality") of result sets that are generated by an RDBMS in response to user queries. Instead, when it is desired to limit the number of tuples in a result set, an application program must request the entire result set from RDBMS and then fetch only the desired number of tuples from the result set.

Thus, in the first example SQL does not support a query that asks for only the ten images that most resemble the input image. In the second example there is no way to submit and execute a request for only the ten highest-grossing videos, ordered by income.

Unfortunately, the above-described shortcoming means that, for any query, the RDBMS must be ready to generate an entire result set, even though only a subset of the query result is desired, with the application program simply trimming the result set down to the desired size. Thus, for the first example above, every image in the database that satisfied a particular SQL-generated query plan, and not just the "closest" ten images, would likely be processed into the result set. Likewise, in the second example, the entire set of revenue-generating videos in the database would be processed into the result set.

Means are available in SQL for implicity limiting result set size, for example, by limit testing. However, the efficacy of this technique depends upon making a good, informed guess as to how a specified limit divides the set of all possible results. SQL also provides means to limit the size of a subset of results that are returned to a user by specifying the size of a set of result items transferred from the RDBMS to the user from a result set generated by the RDBMS in response to the user's query. Neither of these SQL mechanisms, however, provides an explicit limit on the size of a result set that may be declared in a query.

As a consequence of entirely processing a query even when only a limited response is necessary, the response time of the RDBMS to execute the query can be prolonged unnecessarily, computing time wasted, and processing capacity reduced for concurrently executing multiple queries. Furthermore, the continuing growth in size and usage of relational databases compounds these problems. Moreover, by not incorporating a cardinality limitation in query results, an RDBMS cannot exploit additional cardinality-related information that otherwise would be available to it.

SUMMARY OF THE INVENTION

The present invention recognizes the desirability of incorporating an explicit cardinality-limiting operator in a database language that enables a user to limit the size of a query result. In addition, the present invention understands that any modification to existing database management systems that use SQL preferably requires a minimal change to the database management systems themselves. In other words, it is desirable that the specification of a cardinality limit in SQL require at most minimal or no changes to other operations of a database management system, such as join and merge.

Accordingly, it is an object of the present invention to provide a method and apparatus for explicitly limiting the

US005956706A

United States Patent [19]

Carey et al.

[11] Patent Number: 5,956,706

[45] Date of Patent: Sep. 21, 1999

[54] METHOD AND SYSTEM FOR LIMITING THE CARDINALITY OF AN SQL QUERY RESULT

[75] Inventors: Michael James Carey, San Jose, Calif.; Donald Alan Kossmann, Passau, Germany

[73] Assignee: International Business Machines Corporation, Armonk, N.Y.

[21] Appl. No.: 08/853,685

[22] Filed: May 9, 1997

[51] Int. Cl.^{6} G06F 17/30

[52] U.S. Cl. 707/2

[58] Field of Search 707/1–10, 100–104, 707/200–206

[56] References Cited

U.S. PATENT DOCUMENTS

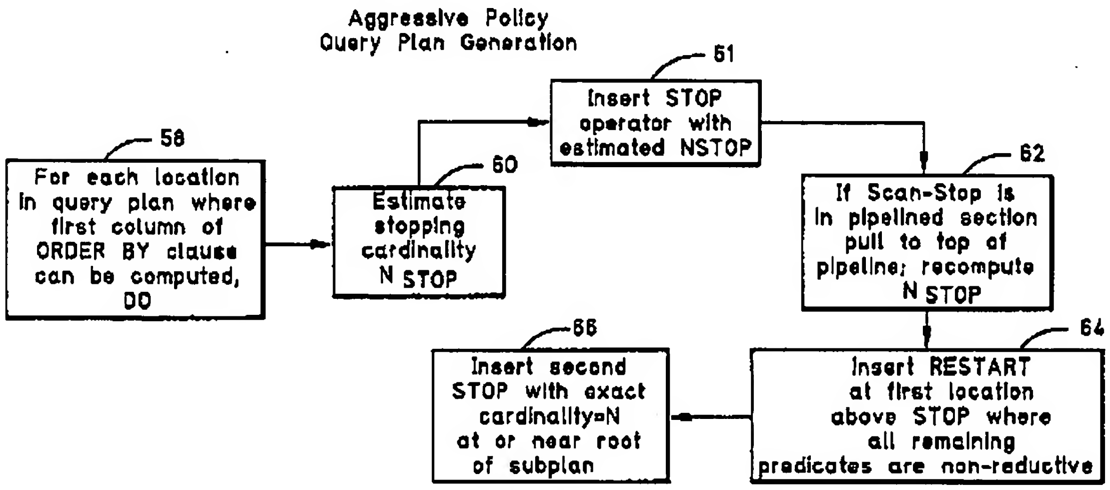
4,769,772	9/1988	Dwyer	707/2
5,367,675	11/1994	Chen et al.	395/600
5,412,804	5/1995	Krishna	395/600
5,412,806	5/1995	Du et al.	707/2
5,530,939	6/1996	Mansfiled, Jr. et al.	395/600
5,546,570	8/1996	McPherson, Jr. et al.	395/600
5,548,754	8/1996	Pirahesh et al.	395/600
5,548,758	8/1996	Pirahesh et al.	395/600
5,598,559	1/1997	Chaudhuri	707/2
5,794,228	8/1998	French et al.	707/2
5,794,229	8/1998	French et al.	707/2

Primary Examiner—Ruay Lian Ho
Attorney, Agent, or Firm—Gray Cary Ware Freidenrich

[57] ABSTRACT

A STOP AFTER clause for a relational database management system (RDBMS) structured query language (SQL) causes the RDBMS, in response to a query, to return a query result having at most a predetermined cardinality specified in the query. A query with a STOP AFTER clause is processed by the RDBMS by provision of one or more STOP operators that are inserted into a query execution plan generated by the RDBMS to execute the query. In a conservative policy, STOP operators are inserted in the query execution plan such that no tuples that might be required in the query result are discarded. In contrast, an aggressive policy inserts a STOP operator in the query execution plan wherever it is able to provide a beneficial cardinality reduction. A RESTART operator is inserted into aggressive policy query execution plans to ensure that at least the predetermined number of tuples are returned in the query result, and a final STOP operator is added at or near the root of the plan to ensure that at most the specified number of tuples are produced.

25 Claims, 4 Drawing Sheets


Aggressive Policy
Query Plan Generation
58
For each location in query plan where first column of ORDER BY clause can be computed, DO
60
Estimate stopping cardinality N STOP
61
Insert STOP operator with estimated NSTOP
62
If Scan-Stop is in pipelined section pull to top of pipeline; recompute N STOP
64
Insert RESTART at first location above STOP where all remaining predicates are non-reductive
66
Insert second STOP with exact cardinality=N at or near root of subplan

1/24/05, EAST Version: 2.0.1.4